A-Scan2BIM: Assistive Scan to Building Information Modeling Supplementary Material

The supplementary document provides (Section 1) details of our plugin that records Revit modeling sequences; (Section 2) architecture specification for the next wall predictor; and (Section 3) architecture/training details for TCN used in our novel ordering metric. Please also refer to the supplementary video demonstrating our dataset and assistive modeling system in action.

1 Revit recorder plugin

To record architect modeling sequence, we created a custom Revit plugin, as Revit does not natively provide this functionality. The plugin is written in C#, and is triggered after each addition/modification/deletion of elements through an event handler. Upon triggering, the full state of the affected elements is accessible through the Revit API and saved. By exploiting C#'s Reflection functionality, we can then automatically iterate through the element object's properties and write them to a JSON file. We also modified the auto-save functionality of Revit so we save the state of the model every 15 minutes. The JSON files are also split into 15-minute sessions as a fail-safe in case of any errors during recording.

To obtain an equivalent API call that mimics a modeling operation, we find the difference between the saved element states, and translate it programmatically using the Revit API. A second custom Revit plugin is created for the purpose of replaying the recorded data step-by-step, which is also demonstrated in the video.

2 Next wall predictor architecture

Figure 1 shows the high-level architecture of our next wall predictor. Each wall node contains three information: the wall endpoint coordinates (x_0, y_0, x_1, y_1), its timestep t, and one of three wall type encodings. The number in parenthesis in the green boxes denotes feature dimension, and the two outer linear layers map dimensions from $512 \rightarrow 256$ and $256 \rightarrow 256$ respectively. The Transformer network is a stack of six blocks, with each block consisting of a mix of multi-head attention, dropout, layer normalization, linear, and ReLU activation layers. The transformer block is similar to that of the one proposed in [2], except that we removed any modules related to deformable image attention as our prediction network relies on geometry only. In the figure we provide the full details for ease of reproducibility.

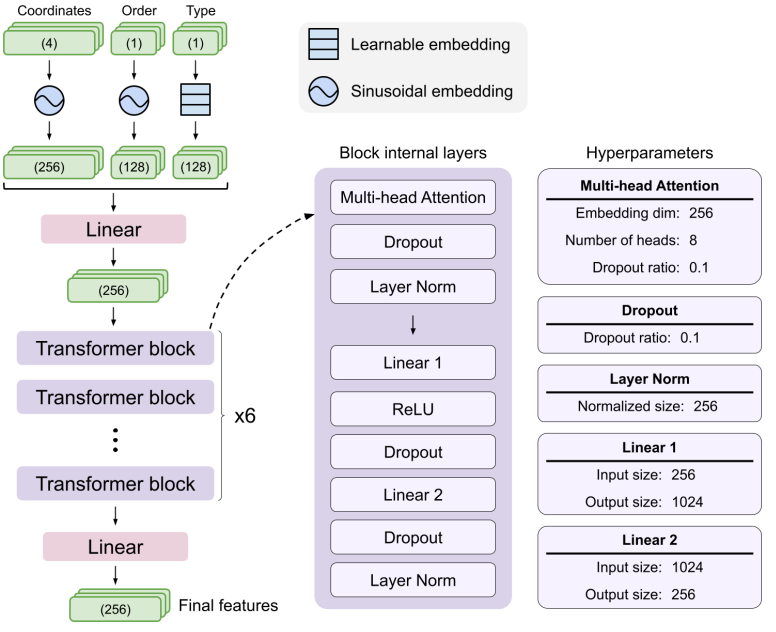

Figure 1: Next wall predictor architecture. Note that our transformer block is similar to that of [2], except without the deformable image attention module.

3 TCN architecture and training details

To obtain a latent encoding for an arbitrary modeling sequences, we train a Temporal Convolutional Network (TCN) to auto-encode wall coordinates and extract hidden features from a middle layer. Specifically, we borrow the implementation from [1], and use the code for the "Copy Memory Task". During training, we construct an example by sampling between 2 to 10 edges from a random floor. We flatten the wall coordinates to a one-dimensional vector as network input and use the L2 distance as the reconstruction loss. Batch size of 32 and Adam optimizer is used with learning rate $5e^{-4}$, and we train until convergence, around 20000 iterations. During inference, we flatten the features after the 6^{th} layer as the sequence encoding to be used for the order metric.

References

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271*, 2018.

[2] Jiacheng Chen, Yiming Qian, and Yasutaka Furukawa. HEAT: Holistic Edge Attention Transformer for Structured Reconstruction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3856–3865. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.00384. URL https://ieeexplore.ieee.org/document/9878511/.